Projekt 1

do samodzielnego wykonania

Zweryfikować przedstawioną na wykładzie ocenę średniej i pesymistycznej złożoności wyszukiwania liniowego i binarnego.

Przeprowadzić analizę za pomocą instrumentacji i pomiarów czasu. W porównaniu wykorzystać tablice liczb całkowitych o rozmiarze rzędu 2^{30} bajtów (2^{28} elementów typu *uint/int*).

W sprawozdaniu przedstawić dla każdego algorytmu:

- kod źródłowy przed instrumentacją
- kod źródłowy po instrumentacji
- zebrane wyniki w postaci tekstu i wykresów
- wnioski z analizy zebranych danych

Wyszukiwanie Liniowe Średnie

Funkcja wyszukiwania liniowego średniego przed Instrumentacją

```
static int WyszukiwanieLiniowePrzedInstrumentacja(int[] Tablica2, int liczba2)//wyszukiwanie liniowe bez instrumentacji
{

    for (int i = 0; i < Tablica2.Length; i++)
    {


        if (Tablica2[i] == liczba2)
        {
            return i;
        }
    }
    return -1;

}
```

Funkcja wyszukiwania liniowego średniego z instrumentacją

```
static int WyszukiwanieLiniowePoInstrumentacji(int[] Tablica2, int liczba2)//wyszukiwanie liniowe z instrumentacją
{
    Suma = 0;
    Licznik = 0;
    for (int i = 0; i < Tablica2.Length; i++)
    {
        ++Licznik;
        Suma += Tablica2[i];
        Wynik = Suma / Licznik;

        if (Tablica2[i] == liczba2)
        {
            return i;
        }
    }
    return -1;

}
```

Funkcje zaimplementowane w Main

```
static void Main(string[] args)
{
    long czasroznica;
    int indeks = 0;
    Console.WriteLine("LinioweŚrednie");
    Console.WriteLine("NR ; Rozmiar ; Szukana ; Nr indeksu ; Ilość operacji  ; Czas       ; Złożoność");

    for (int j = 4473924; j < (int)Math.Pow(2, 28); j += 4473924)
    {
        NR++;
        int SzukanaLiczba;


        int[] Tablica = new int[j];
        for (int i = 0; i < Tablica.Length; i++)
        {
            Tablica[i] = i + 1;

        }
        Array.Sort(Tablica);
        SzukanaLiczba = Tablica.Length / 2;

        var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
        czas.Start();//czas start
        indeks = WyszukiwanieLiniowePrzedInstrumentacja(Tablica, SzukanaLiczba);//Funkcja liniowa bez instrumentacji
        czas.Stop();//czas stop
        czasroznica = czas.ElapsedTicks;//obliczanie czasu w tikach procesora

        indeks = WyszukiwanieLiniowePoInstrumentacji(Tablica, SzukanaLiczba);
        Console.WriteLine(NR + " ; " + Tablica.Length + " ; " + SzukanaLiczba + " ; " + indeks + "    ; " + Licznik + "        ; " + czas.ElapsedTicks + " ; " + Wynik);


    }
    Console.ReadKey();
}
```

Obliczanie czasu operacji bez instrumentacji

```
var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
czas.Start();//czas start
indeks = WyszukiwanieLiniowePrzedInstrumentacja(Tablica, SzukanaLiczba);//Funkcja liniowa bez instrumentacji
czas.Stop();//czas stop
czasroznica = czas.ElapsedTicks;//obliczanie czasu w tikach procesora
```

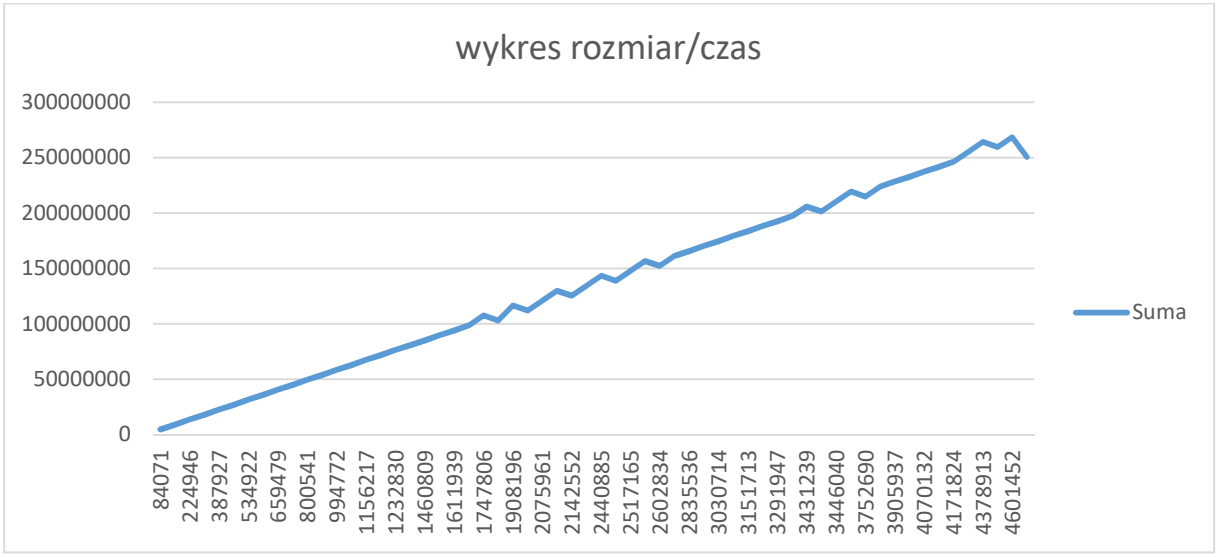
Pomiary:

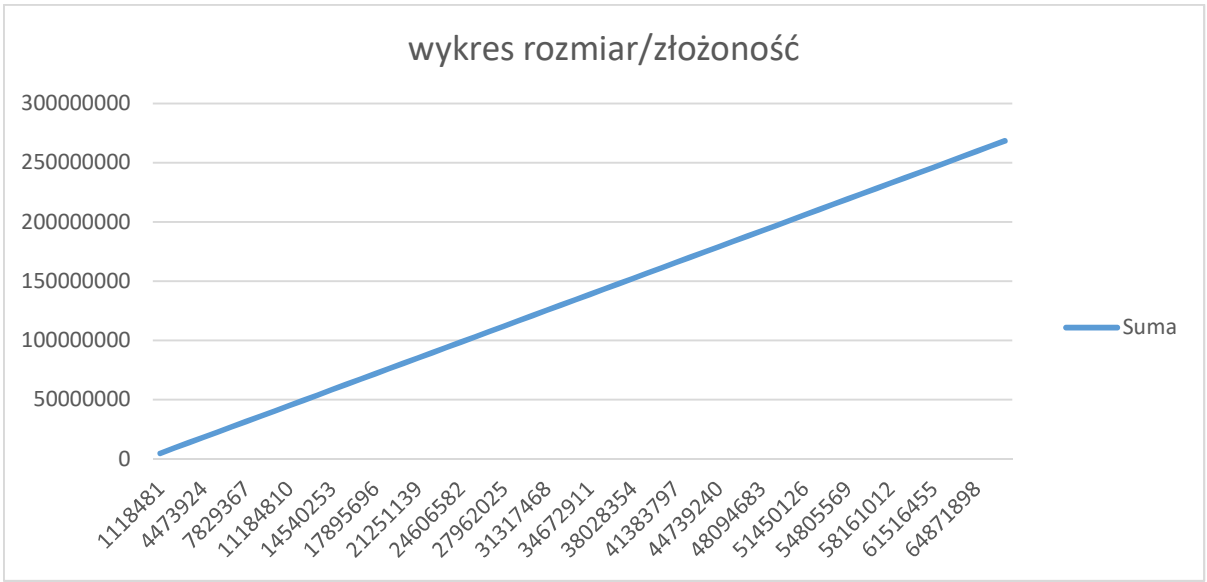
NR	Rozmiar	Szukana	Nr indeksu	Ilość operacji	Czas	Złożoność
1	4473924	2236962	2236961	2236962	84071	1118481
2	8947848	4473924	4473923	4473924	153208	2236962
3	13421772	6710886	6710885	6710886	224946	3355443
4	17895696	8947848	8947847	8947848	302507	4473924
5	22369620	11184810	11184809	11184810	387927	5592405
6	26843544	13421772	13421771	13421772	458514	6710886
7	31317468	15658734	15658733	15658734	534922	7829367
8	35791392	17895696	17895695	17895696	616865	8947848
9	40265316	20132658	20132657	20132658	659479	10066329
10	44739240	22369620	22369619	22369620	725585	11184810
11	49213164	24606582	24606581	24606582	800541	12303291
12	53687088	26843544	26843543	26843544	917244	13421772

13	58161012	29080506	29080505	29080506	994772	14540253
14	62634936	31317468	31317467	31317468	1071876	15658734
15	67108860	33554430	33554429	33554430	1156217	16777215
16	71582784	35791392	35791391	35791392	1172003	17895696
17	76056708	38028354	38028353	38028354	1232830	19014177
18	80530632	40265316	40265315	40265316	1377344	20132658
19	85004556	42502278	42502277	42502278	1460809	21251139
20	89478480	44739240	44739239	44739240	1530706	22369620
21	93952404	46976202	46976201	46976202	1611939	23488101
22	98426328	49213164	49213163	49213164	1688282	24606582
23	102900252	51450126	51450125	51450126	1759476	25725063
24	107374176	53687088	53687087	53687088	1747806	26843544
25	111848100	55924050	55924049	55924050	1917792	27962025
26	116322024	58161012	58161011	58161012	1908196	29080506
27	120795948	60397974	60397973	60397974	2075961	30198987
28	125269872	62634936	62634935	62634936	2142552	31317468
29	129743796	64871898	64871897	64871898	2122083	32435949
30	134217720	67108860	67108859	67108860	2294875	33554430
31	138691644	69345822	69345821	69345822	2456279	34672911
32	143165568	71582784	71582783	71582784	2440885	35791392
33	147639492	73819746	73819745	73819746	2517165	36909873
34	152113416	76056708	76056707	76056708	2602834	38028354
35	156587340	78293670	78293669	78293670	2567591	39146835
36	161061264	80530632	80530631	80530632	2678530	40265316
37	165535188	82767594	82767593	82767594	2835536	41383797
38	170009112	85004556	85004555	85004556	2882814	42502278
39	174483036	87241518	87241517	87241518	3030714	43620759
40	178956960	89478480	89478479	89478480	3068865	44739240
41	183430884	91715442	91715441	91715442	3151713	45857721
42	187904808	93952404	93952403	93952404	3211389	46976202
43	192378732	96189366	96189365	96189366	3291947	48094683
44	196852656	98426328	98426327	98426328	3367371	49213164
45	201326580	100663290	100663289	100663290	3441194	50331645
46	205800504	102900252	102900251	102900252	3431239	51450126
47	210274428	105137214	105137213	105137214	3446040	52568607
48	214748352	107374176	107374175	107374176	3752690	53687088
49	219222276	109611138	109611137	109611138	3622335	54805569
50	223696200	111848100	111848099	111848100	3762779	55924050
51	228170124	114085062	114085061	114085062	3905937	57042531

52	232644048	116322024	116322023	116322024	3988458	58161012
53	237117972	118558986	118558985	118558986	4070132	59279493
54	241591896	120795948	120795947	120795948	4120492	60397974
55	246065820	123032910	123032909	123032910	4171824	61516455
56	250539744	125269872	125269871	125269872	4997758	62634936
57	255013668	127506834	127506833	127506834	4371993	63753417
58	259487592	129743796	129743795	129743796	4459418	64871898
59	263961516	131980758	131980757	131980758	4378913	65990379
60	268435440	134217720	134217719	134217720	4601452	67108860

Wykresy:





Wyszukiwanie Liniowe Pesymistyczne

Funkcja wyszukiwania liniowego pesymistycznego przed Instrumentacją

```
static int WyszukiwanieLiniowePrzedInstrumentacja(int[] Tablica2, int liczba2)//wyszukiwanie liniowe pesymistyczne bez instrumentacji
{

    for (int i = 0; i < Tablica2.Length; i++)
    {


        if (Tablica2[i] == liczba2)
        {
            return i;
        }

    }
    return -1;

}
```

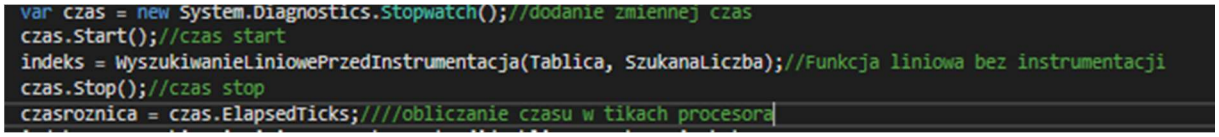
Funkcja wyszukiwania liniowego pesymistycznego z instrumentacją

```
static int WyszukiwanieLiniowePoInstrumentacji(int[] Tablica2, int liczba2)//wyszukiwanie liniowe pesymistyczne z instrumentacją
{

    Licznik = 0;
    for (int i = 0; i < Tablica2.Length; i++)
    {
        ++Licznik;


        if (Tablica2[i] == liczba2)
        {
            return i;
        }

    }
    return -1;

}
```

Funkcje zaimplementowane w Main

```
static void Main(string[] args)
{
    long czasroznica;
    int indeks = 0;
    Console.WriteLine("LiniowePesymistyczne");
    Console.WriteLine("NR ; Rozmiar ; Szukana ; Nr indeksu ; Ilość operacji  ; Czas       ");
    for (int j = 4473924; j < (int)Math.Pow(2, 28); j += 4473924)
    {

        NR++;
        int SzukanaLiczba;

        int[] Tablica = new int[j];
        for (int i = 0; i < Tablica.Length; i++)
        {
            Tablica[i] = i + 1;

        }
        Array.Sort(Tablica);
        SzukanaLiczba = Tablica.Length + 1;
        var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
        czas.Start();//czas start
        indeks = WyszukiwanieLiniowePrzedInstrumentacja(Tablica, SzukanaLiczba);//Funkcja liniowa bez instrumentacji
        czas.Stop();//czas stop
        czasroznica = czas.ElapsedTicks;////obliczanie czasu w tikach procesora
        indeks = WyszukiwanieLiniowePoInstrumentacji(Tablica, SzukanaLiczba);

        Console.WriteLine(NR + " ; " + Tablica.Length + " ; " + SzukanaLiczba + " ; " + indeks + "    ; " + Licznik + "        ; " + czas.ElapsedTicks);


    }
    Console.ReadKey();
}
```

Obliczanie czasu

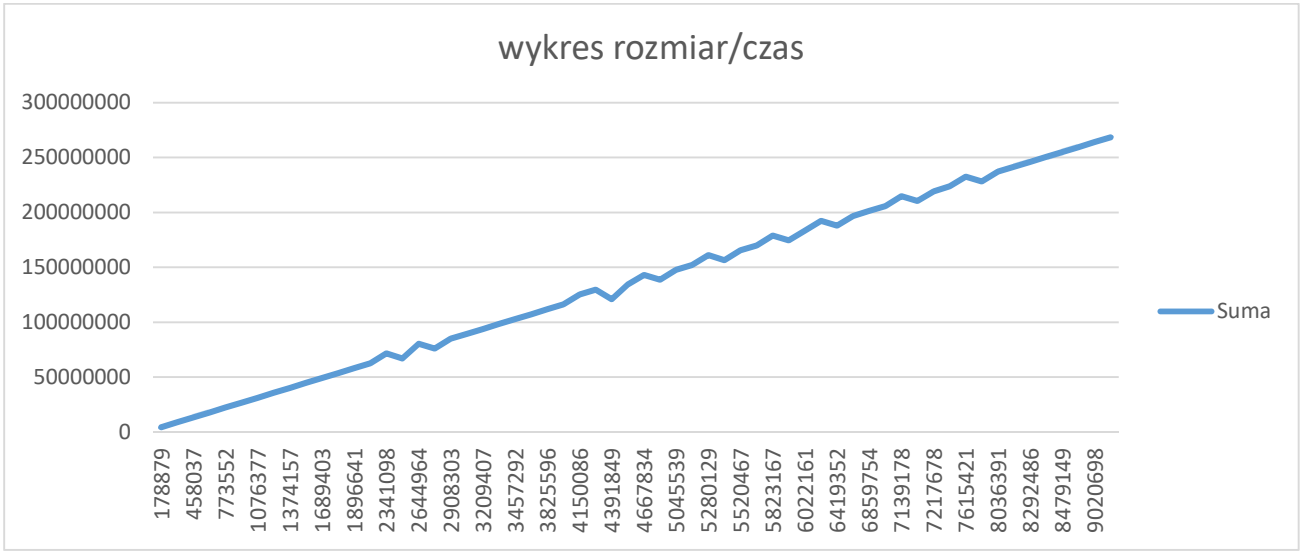


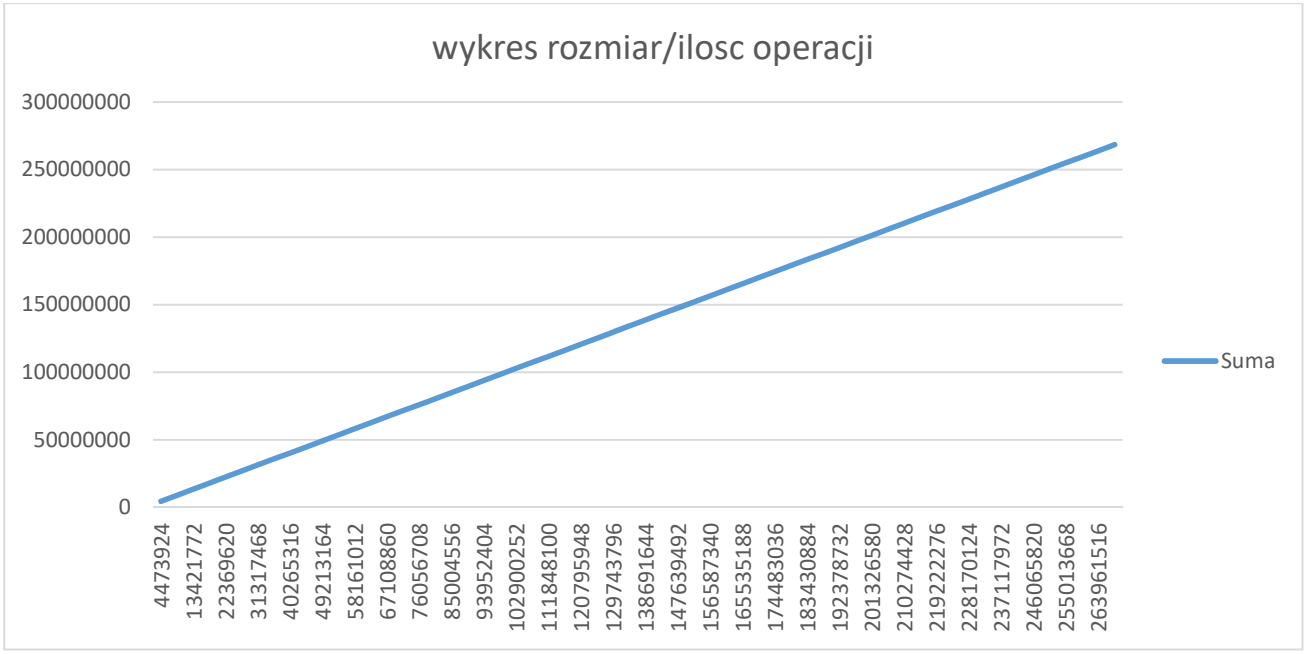
Pomiary:

NR	Rozmiar	Szukana	Nr indeksu	Ilość operacji	Czas
1	4473924	4473925	-1	4473924	178879
2	8947848	8947849	-1	8947848	306250
3	13421772	13421773	-1	13421772	458037
4	17895696	17895697	-1	17895696	629588
5	22369620	22369621	-1	22369620	773552
6	26843544	26843545	-1	26843544	898807
7	31317468	31317469	-1	31317468	1076377
8	35791392	35791393	-1	35791392	1221776
9	40265316	40265317	-1	40265316	1374157
10	44739240	44739241	-1	44739240	1528132
11	49213164	49213165	-1	49213164	1689403
12	53687088	53687089	-1	53687088	1838227
13	58161012	58161013	-1	58161012	1896641
14	62634936	62634937	-1	62634936	2039650
15	67108860	67108861	-1	67108860	2378919
16	71582784	71582785	-1	71582784	2341098
17	76056708	76056709	-1	76056708	2718505
18	80530632	80530633	-1	80530632	2644964
19	85004556	85004557	-1	85004556	2908303
20	89478480	89478481	-1	89478480	3185731
21	93952404	93952405	-1	93952404	3209407
22	98426328	98426329	-1	98426328	3291252
23	102900252	102900253	-1	102900252	3457292
24	107374176	107374177	-1	107374176	3520912
25	111848100	111848101	-1	111848100	3825596
26	116322024	116322025	-1	116322024	3936055
27	120795948	120795949	-1	120795948	4391849
28	125269872	125269873	-1	125269872	4150086

29	129743796	129743797	-1	129743796	4268183
30	134217720	134217721	-1	134217720	4533435
31	138691644	138691645	-1	138691644	4688682
32	143165568	143165569	-1	143165568	4667834
33	147639492	147639493	-1	147639492	5045539
34	152113416	152113417	-1	152113416	5198009
35	156587340	156587341	-1	156587340	5346487
36	161061264	161061265	-1	161061264	5280129
37	165535188	165535189	-1	165535188	5520467
38	170009112	170009113	-1	170009112	5792505
39	174483036	174483037	-1	174483036	5831578
40	178956960	178956961	-1	178956960	5823167
41	183430884	183430885	-1	183430884	6022161
42	187904808	187904809	-1	187904808	6419352
43	192378732	192378733	-1	192378732	6327047
44	196852656	196852657	-1	196852656	6433674
45	201326580	201326581	-1	201326580	6859754
46	205800504	205800505	-1	205800504	6906855
47	210274428	210274429	-1	210274428	7204026
48	214748352	214748353	-1	214748352	7139178
49	219222276	219222277	-1	219222276	7217678
50	223696200	223696201	-1	223696200	7415098
51	228170124	228170125	-1	228170124	7796106
52	232644048	232644049	-1	232644048	7615421
53	237117972	237117973	-1	237117972	8036391
54	241591896	241591897	-1	241591896	8145492
55	246065820	246065821	-1	246065820	8292486
56	250539744	250539745	-1	250539744	8316607
57	255013668	255013669	-1	255013668	8479149
58	259487592	259487593	-1	259487592	8837938
59	263961516	263961517	-1	263961516	9020698
60	268435440	268435441	-1	268435440	9206948

Wykresy:


wykres rozmiar/czas
300000000
250000000
200000000
150000000
100000000
50000000
0
178879
458037
773552
1076377
1374157
1689403
1896641
2341098
2644964
2908303
3209407
3457292
3825596
4150086
4391849
4667834
5045539
5280129
5520467
5823167
6022161
6419352
6859754
7139178
7217678
7615421
8036391
8292486
8479149
9020698
Suma


wykres rozmiar/ilosc operacji
300000000
250000000
200000000
150000000
100000000
50000000
0
4473924
13421772
22369620
31317468
40265316
49213164
58161012
67108860
76056708
85004556
93952404
102900252
111848100
120795948
129743796
138691644
147639492
156587340
165535188
174483036
183430884
192378732
201326580
210274428
219222276
228170124
237117972
246065820
255013668
263961516
Suma

Wyszukiwanie Binarne Średnie

Funkcja wyszukiwania binarnego średniego przed instrumentacją

```
static int WyszukiwanieBinarnePrzedInstrumentacja(int[] Tablica2, int liczba2, int TablicaDlugosc)//wyszukiwanie binarne bez instrumentacji
{
    int prawa = TablicaDlugosc - 1;
    int lewa = 0;
    int srodek;

    while (lewa <= prawa)
    {


        srodek = (lewa + prawa) / 2;

        if (Tablica2[srodek] == liczba2)
        {

            return srodek;


        }
        else if (Tablica2[srodek] < liczba2)
        {
            lewa = srodek + 1;


        }
        else
        {
            prawa = srodek - 1;


        }



    }

    return -1;
```

Funkcja wyszukiwania binarnego średniego z instrumentacją

```
static int WyszukiwanieBinarnePoInstrumentacji(int[] Tablicabin, int liczba2, int TablicaDlugosc)//wyszukiwanie binarne z instrumentacją
{
    int prawa = TablicaDlugosc - 1;
    int lewa = 0;
    int srodek;
    Licznik = 0;
    Dluga = 0;
    Wynik = 0;
    while (lewa <= prawa)
    {
        Licznik++;

        srodek = (lewa + prawa) / 2;
        Wynik += (ulong)Licznik * (ulong)Math.Pow(2, Licznik - 1);
        Dluga += (ulong)Math.Pow(2, Licznik - 1);
        if (Tablicabin[srodek] == liczba2)
        {
            Licznik++;
            Wynik += (ulong)Licznik * (ulong)Math.Pow(2, Licznik - 1);
            Dluga += (ulong)Math.Pow(2, Licznik - 1);
            Wynik = Wynik / Dluga;

            return srodek;

        }
        else if (Tablicabin[srodek] < liczba2)
        {
            Licznik++;
            Wynik += (ulong)Licznik * (ulong)Math.Pow(2, Licznik - 1);
            Dluga += (ulong)Math.Pow(2, Licznik - 1);
            lewa = srodek + 1;

        }
        else
        {
            Licznik++;
            Wynik += (ulong)Licznik * (ulong)Math.Pow(2, Licznik - 1);
            Dluga += (ulong)Math.Pow(2, Licznik - 1);
            prawa = srodek - 1;

        }

    }
    return -1;
```

Funkcje zaimplementowane w Main

```
static void Main(string[] args)
{
    int indeks = 0;
    long czasroznica;
    Console.WriteLine("Binarne średnie");
    Console.WriteLine("NR ; Rozmiar ; Szukana ; Nr indeksu ; Ilość operacji  ; Czas       ; Złożoność");
    for (int j = 4473924; j < (int)Math.Pow(2, 28); j += 4473924)
    {
        NR++;
        int SzukanaLiczba;
        int Nrindeksu;

        int[] Tablica = new int[j];
        for (int i = 0; i < Tablica.Length; i++)
        {
            Tablica[i] = i + 1;
            Wynik += (ulong)Tablica[i];
        }
        Array.Sort(Tablica);
        SzukanaLiczba = j - 1;
        Nrindeksu = j - 1;


        var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
        czas.Start();//czas start
        indeks = WyszukiwanieBinarnePrzedInstrumentacja(Tablica, SzukanaLiczba, Tablica.Length);//Funkcja liniowa bez instrumentacji
        czas.Stop();//czas stop
        czasroznica = czas.ElapsedTicks;//obliczanie czasu w tikach procesora

        indeks = WyszukiwanieBinarnePoInstrumentacji(Tablica, SzukanaLiczba, Tablica.Length);

        Console.WriteLine(NR + " ; " + Tablica.Length + " ; " + SzukanaLiczba + " ; " + indeks + "    ; " + Licznik + "        ; " + czas.ElapsedTicks + " ; " + Wynik);


    }
    Console.ReadKey();
}
```

Obliczanie czasu

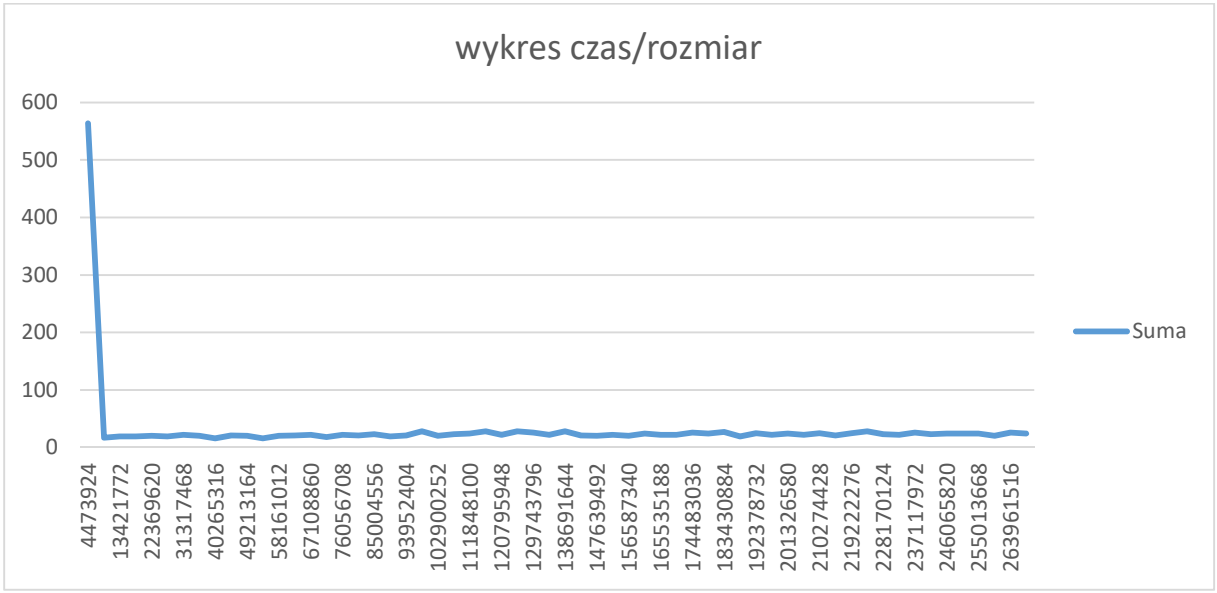
```
var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
czas.Start();//czas start
indeks = WyszukiwanieBinarnePrzedInstrumentacja(Tablica, SzukanaLiczba, Tablica.Length);//Funkcja liniowa bez instrumentacji
czas.Stop();//czas stop
czasroznica = czas.ElapsedTicks;//obliczanie czasu w tikach procesora
```

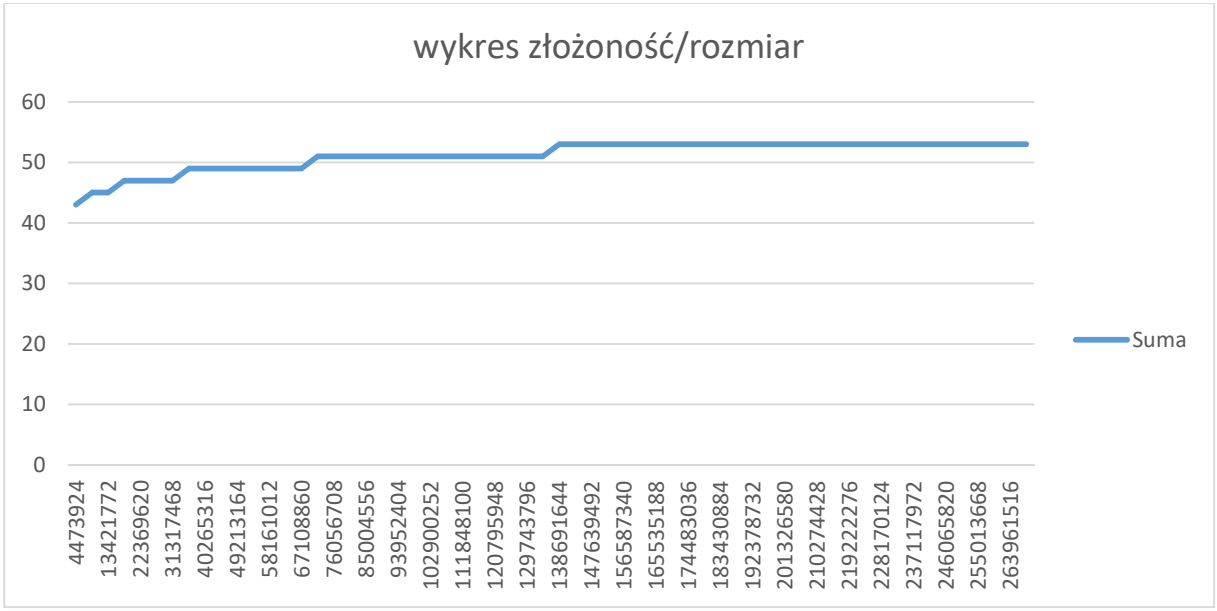
Pomiary:

NR	Rozmiar	Szukana	Nr indeksu	Ilość operacji	Czas	Złożoność
1	4473924	4473923	4473922	44	564	43
2	8947848	8947847	8947846	46	17	45
3	13421772	13421771	13421770	46	19	45
4	17895696	17895695	17895694	48	19	47
5	22369620	22369619	22369618	48	20	47
6	26843544	26843543	26843542	48	19	47
7	31317468	31317467	31317466	48	22	47
8	35791392	35791391	35791390	50	20	49
9	40265316	40265315	40265314	50	16	49
10	44739240	44739239	44739238	50	21	49
11	49213164	49213163	49213162	50	20	49
12	53687088	53687087	53687086	50	16	49
13	58161012	58161011	58161010	50	20	49
14	62634936	62634935	62634934	50	21	49
15	67108860	67108859	67108858	50	22	49
16	71582784	71582783	71582782	52	18	51
17	76056708	76056707	76056706	52	22	51
18	80530632	80530631	80530630	52	21	51
19	85004556	85004555	85004554	52	23	51
20	89478480	89478479	89478478	52	19	51
21	93952404	93952403	93952402	52	21	51
22	98426328	98426327	98426326	52	28	51
23	102900252	102900251	102900250	52	20	51
24	107374176	107374175	107374174	52	23	51
25	111848100	111848099	111848098	52	24	51
26	116322024	116322023	116322022	52	28	51
27	120795948	120795947	120795946	52	22	51
28	125269872	125269871	125269870	52	28	51
29	129743796	129743795	129743794	52	26	51

30	134217720	134217719	134217718	52	22	51
31	138691644	138691643	138691642	54	28	53
32	143165568	143165567	143165566	54	21	53
33	147639492	147639491	147639490	54	20	53
34	152113416	152113415	152113414	54	22	53
35	156587340	156587339	156587338	54	20	53
36	161061264	161061263	161061262	54	24	53
37	165535188	165535187	165535186	54	22	53
38	170009112	170009111	170009110	54	22	53
39	174483036	174483035	174483034	54	26	53
40	178956960	178956959	178956958	54	24	53
41	183430884	183430883	183430882	54	27	53
42	187904808	187904807	187904806	54	19	53
43	192378732	192378731	192378730	54	25	53
44	196852656	196852655	196852654	54	22	53
45	201326580	201326579	201326578	54	24	53
46	205800504	205800503	205800502	54	22	53
47	210274428	210274427	210274426	54	25	53
48	214748352	214748351	214748350	54	21	53
49	219222276	219222275	219222274	54	25	53
50	223696200	223696199	223696198	54	28	53
51	228170124	228170123	228170122	54	23	53
52	232644048	232644047	232644046	54	22	53
53	237117972	237117971	237117970	54	26	53
54	241591896	241591895	241591894	54	23	53
55	246065820	246065819	246065818	54	24	53
56	250539744	250539743	250539742	54	24	53
57	255013668	255013667	255013666	54	24	53
58	259487592	259487591	259487590	54	20	53
59	263961516	263961515	263961514	54	26	53
60	268435440	268435439	268435438	54	24	53

Wykresy:


wykres czas/rozmiar
600
500
400
300
200
100
0
4473924
13421772
22369620
31317468
40265316
49213164
58161012
67108860
76056708
85004556
93952404
102900252
111848100
120795948
129743796
138691644
147639492
156587340
165535188
174483036
183430884
192378732
201326580
210274428
219222276
228170124
237117972
246065820
255013668
263961516
Suma


wykres złożoność/rozmiar
60
50
40
30
20
10
0
4473924
13421772
22369620
31317468
40265316
49213164
58161012
67108860
76056708
85004556
93952404
102900252
111848100
120795948
129743796
138691644
147639492
156587340
165535188
174483036
183430884
192378732
201326580
210274428
219222276
228170124
237117972
246065820
255013668
263961516
Suma

Wyszukiwanie Binarne Pesymistyczne

Funkcja wyszukiwania binarnego pesymistycznego przed instrumentacją

```
static int WyszukiwanieBinarnePrzedInstrumentacja(int[] Tablica2, int liczba2, int tablicadlugosc)//wyszukiwanie binarne pesymistyczne bez instrumentacji
{
    int prawa = tablicadlugosc - 1;
    int lewa = 0;
    int srodek;

    while (lewa <= prawa)
    {


        srodek = (lewa + prawa) / 2;

        if (Tablica2[srodek] == liczba2)
        {

            return srodek;


        }
        else if (Tablica2[srodek] < liczba2)
        {
            lewa = srodek + 1;


        }
        else
        {
            prawa = srodek - 1;


        }



    }

    return -1;
```

Funkcja wyszukiwania binarnego pesymistycznego z instrumentacją

```
static int WyszukiwanieBinarnePoInstrumentacji(int[] Tablica2, int liczba2, int tablicadlugosc)//wyszukiwanie binarne pesymistyczne z instrumentacją
{
    int prawa = tablicadlugosc - 1;
    int lewa = 0;
    int srodek;
    Licznik = 0;

    while (lewa <= prawa)
    {
        ++Licznik;

        srodek = (lewa + prawa) / 2;

        if (Tablica2[srodek] == liczba2)
        {
            ++Licznik;

            return srodek;


        }
        else if (Tablica2[srodek] < liczba2)
        {
            lewa = srodek + 1;
            ++Licznik;


        }
        else
        {
            prawa = srodek - 1;
            ++Licznik;


        }

    }
    return -1;

}
```

Funkcje zaimplementowane w Main

```
static void Main(string[] args)
{

    int indeks = 0;
    long czasroznica;
    Console.WriteLine("Binarne pesymistyczne");
    Console.WriteLine("NR ; Rozmiar ; Szukana ; Nr indeksu ; Ilość operacji  ; Czas      ");

    for (int j = 4473924; j < (int)Math.Pow(2, 28); j += 4473924)
    {
        NR++;
        int SzukanaLiczba;


        int[] Tablica = new int[j];
        for (int i = 0; i < Tablica.Length; i++)
        {
            Tablica[i] = i + 1;
        }
        Array.Sort(Tablica);
        SzukanaLiczba = j + 1;


        var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
        czas.Start();//czas start
        indeks = WyszukiwanieBinarnePrzedInstrumentacja(Tablica, SzukanaLiczba, Tablica.Length);//Funkcja liniowa bez instrumentacji
        czas.Stop();//czas stop
        czasroznica = czas.ElapsedTicks; ;//obliczanie czasu w tikach procesora

        indeks = WyszukiwanieBinarnePoInstrumentacji(Tablica, SzukanaLiczba, Tablica.Length);

        Console.WriteLine(NR + " ; " + Tablica.Length + " ; " + SzukanaLiczba + " ; " + indeks + "    ; " + Licznik + "       ; " + czas.ElapsedTicks);

    }
    Console.ReadKey();
}
```

Obliczanie czasu

```
var czas = new System.Diagnostics.Stopwatch();//dodanie zmiennej czas
czas.Start();//czas start
indeks = WyszukiwanieBinarnePrzedInstrumentacja(Tablica, SzukanaLiczba, Tablica.Length);//Funkcja liniowa bez instrumentacji
czas.Stop();//czas stop
czasroznica = czas.ElapsedTicks; ;//obliczanie czasu w tikach procesora
```

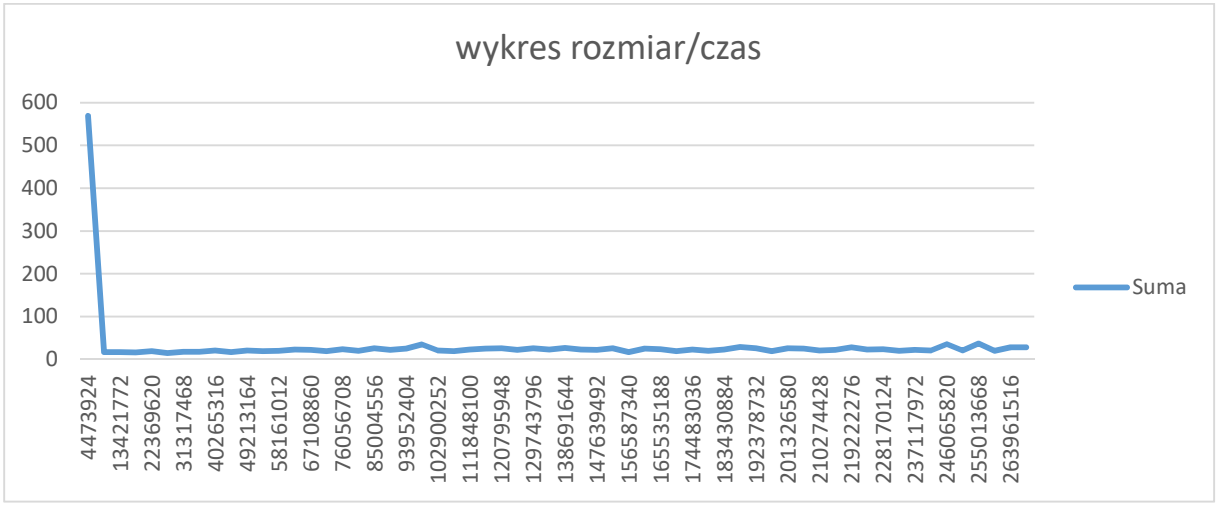
Pomiary:

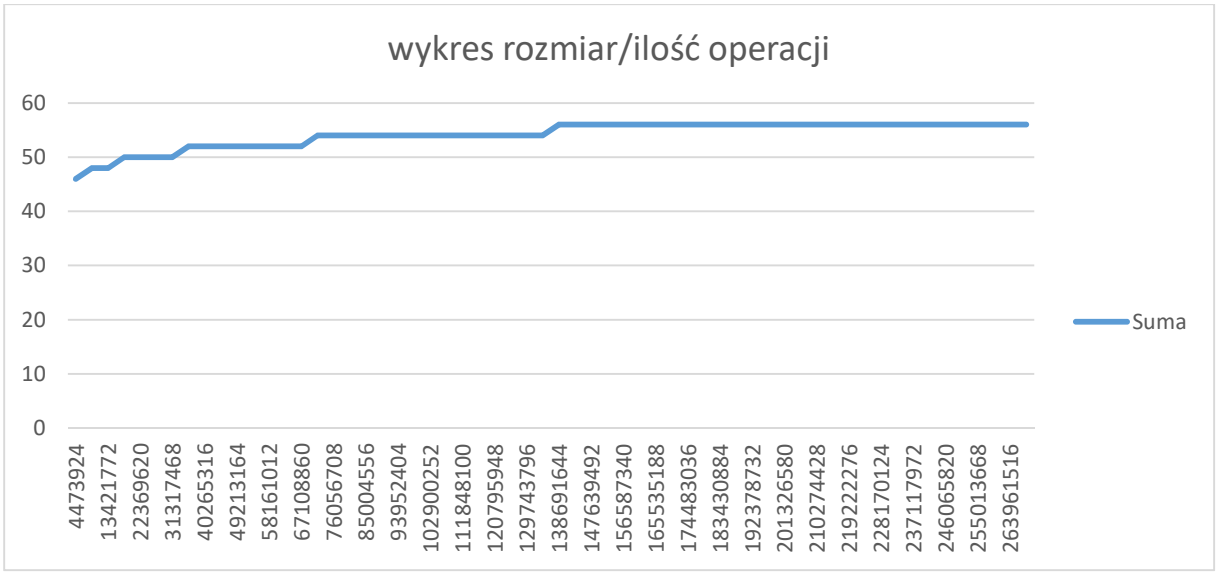
NR	Rozmiar	Szukana	Nr indeksu	Ilość operacji	Czas
1	4473924	4473925	-1	46	569
2	8947848	8947849	-1	48	17
3	13421772	13421773	-1	48	17
4	17895696	17895697	-1	50	16
5	22369620	22369621	-1	50	19
6	26843544	26843545	-1	50	15
7	31317468	31317469	-1	50	18
8	35791392	35791393	-1	52	18
9	40265316	40265317	-1	52	21
10	44739240	44739241	-1	52	17
11	49213164	49213165	-1	52	21

12	53687088	53687089	-1	52	19
13	58161012	58161013	-1	52	20
14	62634936	62634937	-1	52	23
15	67108860	67108861	-1	52	22
16	71582784	71582785	-1	54	19
17	76056708	76056709	-1	54	24
18	80530632	80530633	-1	54	20
19	85004556	85004557	-1	54	26
20	89478480	89478481	-1	54	22
21	93952404	93952405	-1	54	25
22	98426328	98426329	-1	54	35
23	102900252	102900253	-1	54	21
24	107374176	107374177	-1	54	19
25	111848100	111848101	-1	54	23
26	116322024	116322025	-1	54	25
27	120795948	120795949	-1	54	26
28	125269872	125269873	-1	54	22
29	129743796	129743797	-1	54	26
30	134217720	134217721	-1	54	23
31	138691644	138691645	-1	56	27
32	143165568	143165569	-1	56	23
33	147639492	147639493	-1	56	22
34	152113416	152113417	-1	56	26
35	156587340	156587341	-1	56	17
36	161061264	161061265	-1	56	25
37	165535188	165535189	-1	56	24
38	170009112	170009113	-1	56	19
39	174483036	174483037	-1	56	23
40	178956960	178956961	-1	56	20
41	183430884	183430885	-1	56	23
42	187904808	187904809	-1	56	29
43	192378732	192378733	-1	56	26
44	196852656	196852657	-1	56	19
45	201326580	201326581	-1	56	26
46	205800504	205800505	-1	56	25
47	210274428	210274429	-1	56	21
48	214748352	214748353	-1	56	22
49	219222276	219222277	-1	56	28
50	223696200	223696201	-1	56	23

51	228170124	228170125	-1	56	24
52	232644048	232644049	-1	56	20
53	237117972	237117973	-1	56	22
54	241591896	241591897	-1	56	21
55	246065820	246065821	-1	56	36
56	250539744	250539745	-1	56	21
57	255013668	255013669	-1	56	37
58	259487592	259487593	-1	56	20
59	263961516	263961517	-1	56	28
60	268435440	268435441	-1	56	28

Wykresy:





Podsumowanie:

Wyszukiwanie liniowe polega na przeglądaniu kolejnych elementów tablicy. W przypadku pesymistycznym, gdy poszukiwanego zbioru nie ma w zbiorze lub znajduje się on na samym końcu zbioru, algorytm musi wykonać przynajmniej *n* obiegów pętli sprawdzającej poszczególne elementy. Wynika z tego że pesymistyczna złożoność obliczeniowa jest równa O(n), czyli jest liniowa.

Wyszukiwanie binarne algorytm szuka danego elementu w tablicy uporządkowanej (posortowanej). Złożoność obliczeniowa tego sposobu wyszukiwania jest rzędu O(log). Oznacza to, że metoda jest znaczenie szybsza niż algorytm przeszukiwania liniowego.

Czas potrzebny na wykonanie wyszukiwania liniowego jest dłuższy niż wyszukiwania binarnego.

Komputer na którym wykonywano pomiary:

Procesor:	Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz 2.50 GHz
Zainstalowana pamięć (RAM):	8,00 GB (dostępne: 7,89 GB)
Typ systemu:	64-bitowy system operacyjny, procesor x64